

А.А. Баркалов, Л.А. Титаренко, С.А. Цололо, А.Н. Мирошкин

Уменьшение числа *LUT*-элементов в схеме автомата Мура

Предложен метод синтеза автомата Мура, позволяющий уменьшить число *LUT*-элементов в схеме формирования функций возбуждения триггеров памяти. Метод основан на наличии свободных выходов встроенных блоков памяти, используемых для реализации системы микроопераций автомата. Приведен пример использования предложенного метода.

The method of the Moore FSM synthesis is proposed. It allows to reduce the *LUT* number, the elements in the circuit of forming the function of the trigger stimulation memory. The method is based on the availability EMB free outputs, that are used for the achievement of the microoperation Moore FSM system. An example of the method application is given.

Запропоновано метод синтезу автомата Мура, який дозволяє зменшити число *LUT*-елементів у схемі формування функцій збудження тригерів пам'яті. Метод засновано на наявності вільних виходів вбудованих блоків пам'яті, які використовуються для реалізації системи мікрооперацій автомата. Наведено приклад застосування запропонованого методу.

Введение. Программируемые логические интегральные схемы (ПЛИС) типа *FPGA* (*field-programmable gate arrays*) широко применяются при реализации сложных цифровых систем [1, 2]. Один из важных блоков цифровых систем – устройство управления [3], реализуемое в виде микропрограммного автомата (МПА) Мура [4]. Характерные особенности МПА Мура – наличие классов псевдоэквивалентных состояний (ПЭС) и регулярный характер системы выходных функций [5], используемые для уменьшения аппаратных затрат в схеме МПА Мура [6].

Современные ПЛИС *FPGA* включают в себя элементы табличного типа *LUT* (*look-up-tables*) и встроенные блоки памяти *Embedded Memory Blocks* (*EMB*) [7, 8]. Элементы *LUT* имеют ограниченное число входов (порядка 6), что вызывает необходимость декомпозиции реализуемых функций [9]. Это в свою очередь приводит к уменьшению быстродействия и увеличению потребляемой мощности [1]. В работах [10, 11] приведены методы оптимизации схем МПА Мура в базе *CPLD* (*complex programmable logic devices*). Эти методы используют наличие большого количества входов (до 30) в макроячейках кристаллов *CPLD*, что позволяет использовать до трех источников кодов состояний, но они не могут быть непосредственно использованы при реализации схем на ПЛИС *FPGA* из-за небольшого числа входов *LUT*-элементов. В статье предлагается модификация одного из методов оптимизации, ориентированных на *CPLD*.

Цель исследований – возможность уменьшения числа *LUT*-элементов двух источников кодов классов ПЭС.

Решаемая в статье задача – разработка метода синтеза МПА Мура, позволяющего уменьшить число *LUT*-элементов в схеме формирования функций возбуждения триггеров памяти. При этом алгоритм управления цифровой системы представляется в виде граф-схемы алгоритма (ГСА) [4].

Реализация автомата Мура на *FPGA*

Пусть алгоритм управления цифровой системы представлен ГСА $\Gamma = \Gamma(B, E)$, где $B = \{b_0, b_E\} \cup E_1 \cup E_2$ – множество вершин, $E = \{ \langle b_q, b_r \rangle \mid b_q, b_r \in B \}$ – множество дуг, b_0 – начальная вершина ГСА, b_E – конечная вершина ГСА, E_1 – множество операторных вершин, E_2 – множество условных вершин. В вершинах $b_q \in E_1$ записываются наборы микроопераций $Y(b_q) \subseteq Y$, где $Y = \{y_1, \dots, y_N\}$ – множество микроопераций операционного автомата цифровой системы [4]. В вершинах $b_q \in E_2$ записываются элементы множества логических условий $X = \{x_1, \dots, x_L\}$. Начальная и конечная вершины ГСА соответствуют состоянию $a_1 \in A = \{a_1, \dots, a_M\}$, где A – множество состояний автомата Мура, а каждая вершина $b_q \in E_1$ соответствует одному из элементов множества A [4]. Логическая схема МПА Мура задается системой уравнений

$$\Phi = \Phi(T, X), \quad (1)$$

$$Y = Y(T), \quad (2)$$

где $T = T\{T_1, \dots, T_R\}$ – множество внутренних переменных, кодирующих состояния $a_m \in A$, $R = \lceil \log_2 M \rceil$; $\Phi = \Phi\{D_1, \dots, D_R\}$ – множество функций возбуждения триггеров памяти состояний. Системы (1)–(2) формируются на основе прямой структурной таблицы (ПСТ) со столбцами: a_m – текущее состояние; $K(a_m)$ – код состояния $a_m \in A$; a_s – состояние перехода; $K(a_s)$ – код состояния $a_s \in A$; X_h – конъюнкция некоторых элементов множества X (или их отрицаний), определяющая переход $\langle a_m, a_s \rangle$; Φ_h – набор функций возбуждения памяти МПА, принимающих единичное значение для переключения памяти из $K(a_m)$ в $K(a_s)$; $h = 1, \dots, H_1(\Gamma)$ – номер строки таблицы. В столбце a_m записывается набор микроопераций $Y(a_m) \subseteq Y$, формируемых в состоянии $a_m \in A$. Естественно, что $Y(a_m) = Y(b_q)$, где вершина $b_q \in E_1$ отмечена состоянием $a_m \in A$.

Как правило, число переходов $H_1(\Gamma)$ больше числа переходов $H_2(\Gamma)$ эквивалентного автомата Мили [4]. Это приводит к увеличению числа *PAL* в схеме МПА Мура в сравнении с этим показателем эквивалентного автомата Мили. Параметр $H_1(\Gamma)$ можно уменьшить наличием ПЭС МПА Мура [5]. Состояния $a_m, a_s \in A$ называются ПЭС, если выходы соответствующих им вершин соединены с входом одной и той же вершины ГСА Γ . Пусть $\Pi_A = \{B_1, \dots, B_I\}$ – разбиение множества A на классы ПЭС ($I \leq M$). Поставим в соответствие классу $B_i \in \Pi_A$ двоичный код $K(B_i)$ разрядности $R_1 = \lceil \log_2 I \rceil$ и используем переменные $\tau_r \in \tau$ для такого кодирования, где $|\tau| = R_1$. В этом случае МПА Мура представляются в виде структуры U_1 (рис. 1). Поскольку МПА Мура U_1 ориентирован на *FPGA*, то в состав схемы входят блоки, состоящие из *LUT*-элементов (*LUTer1*, *LUTer2*) и *EMB* (*EMBer*).

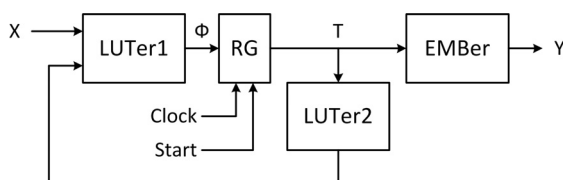


Рис. 1. Структурная схема автомата U_1 , ориентированная на базис ПЛИС *FPGA*

В автомате U_1 блок *LUTer1* реализует систему функций

$$\Phi = \Phi(\tau, X), \quad (3)$$

а блок *EMBer* реализует систему (2). Регистр *RG* представляет собой память состояний, по сигналу *Start* в *RG* заносится нулевой код начального состояния $a_1 \in A$, по сигналу *Clock* происходит смена кодов в регистре. Преобразователь кодов состояний *LUTer2* реализует систему функций

$$\tau = \tau(T), \quad (4)$$

при этом код $K(B_i)$ формируется на основе кода $K(a_m)$, где $a_m \in B_i$.

В работе [10] показано, что для МПА U_1 число переходов уменьшается до $H_2(\Gamma)$. Недостаток модели U_1 – наличие блока *LUTer2*, схема которого потребляет некоторые ресурсы кристалла. В статье предлагается метод синтеза МПА Мура, позволяющий устранить блок *LUTer2*. Предлагаемый метод основан на следующих особенностях микросхем *FPGA* [7, 8]:

- логические элементы состоят из *LUT*, программируемого триггера, мультиплексоров и трехстабильных буферов, что позволяет организовывать мультиплексирование различных *LUT*-элементов;

- встроенные блоки памяти имеют постоянную емкость (V_0), но изменяемое число ячеек (V) и выходов (t_F).

Основная идея предлагаемого метода

Используем идею оптимального кодирования состояний МПА Мура [10], смысл которой заключается в таком кодировании ПЭС, чтобы максимально возможное число классов $B_i \in \Pi_A$ соответствовали одному обобщенному интервалу R -мерного булевого пространства. Представим множество Π_A в виде $\Pi_A = \Pi_B \cup \Pi_C$, где $B_i \in \Pi_B$, если

$$|B_i| > 1, \quad (5)$$

то $B_i \in \Pi_C$ в противном случае. Очевидно, что *LUTer2* должен формировать только коды классов $B_i \in \Pi_B$. Закодируем состояния $a_m \in A$ оптимальным образом [10]. Представим множество Π_B в виде $\Pi_B = \Pi_D \cup \Pi_E$, где $B_i \in \Pi_D$, если коды $a_m \in B$ входят в один обобщенный интервал пространства кодирования. Теперь пре-

образованию подлежат только коды состояний $a_m \in A(\Pi_E)$, где $A(\Pi_E) \subseteq A$ – множество состояний, входящих в классы Π_E . Для кодирования классов $B_i \in \Pi_E$. Достаточно

$$R_2 = \lceil \log_2(|\Pi_E|+1) \rceil \quad (6)$$

переменных, образующих множество Z , где $|Z| = R_2$.

Современные *FPGA* имеют блоки *EMB* с переменной конфигурацией $V \cdot t_F$ [7, 8]. Как правило, поддерживаются следующие конфигурации: $16k \times 1$, $8k \times 2$, $4k \times 4$, $2k \times 8$, $1k \times 16$, 512×32 и 256×64 (бит). Таким образом, параметр t_F принадлежит некоторому множеству $O = \{1, 2, 4, 8, 16, 32, 64\}$.

Параметр t_F для блока *EMBer* определяется как ближайшее целое число из множества O , большее или равное величине

$$t_0 = \lceil V_0/2R \rceil. \quad (7)$$

После определения параметра t_F можно определить общее число выходов t_S в блоках *EMB*, составляющих схему блока *EMBer*:

$$t_S = \lceil N/t_F \rceil \cdot t_F. \quad (8)$$

Очевидно, что может быть Δ_t свободных выходов блоков *EMB*, т.е. выходов, не используемых для представления микроопераций $y_n \in Y$. Это число определяется выражением

$$\Delta_t = t_S - N. \quad (9)$$

Эти выходы можно использовать для представления переменных $z_r \in Z$. Рассмотрим случай, когда выполняется условие

$$\Delta_t \geq R_2 + 1. \quad (10)$$

При выполнении отношения (10) все переменные $z_r \in Z$ реализуются блоком *EMBer*. В отличие от *CPLD* с большим числом входов в макроячейках, *FPGA* включает элементы с очень небольшим числом входов. Поэтому методы [10, 11] не могут быть непосредственно использованы для уменьшения числа *LUT*-элементов. Однако два источника кодов классов ПЭС могут быть использованы и в этом случае, а блок *LUTer1* надо представить в виде двух блоков, выходы которых мультиплексируются.

Пусть блок *LUTerCD* реализует подсистему $\Phi_1 \subseteq \Phi$ для классов $B_i \in (\Pi_C \cup \Pi_D)$:

$$\Phi_1 = \Phi_1(T, X). \quad (11)$$

Пусть блок *LUTerE* реализует подсистему $\Phi_2 \subseteq \Phi$ для классов $B_i \in \Pi_E$:

$$\Phi_2 = \Phi_2(Z, X). \quad (12)$$

Для формирования функций $D_r \in \Phi$ используется мультиплексор *MX*, управляемый переменной y_M :

$$\Phi = y_M \Phi_1 \vee \overline{y_M} \Phi_2. \quad (13)$$

Таким образом, для уменьшения числа *LUT*-элементов в схеме МПА Мура предлагается модель U_2 (рис. 2).

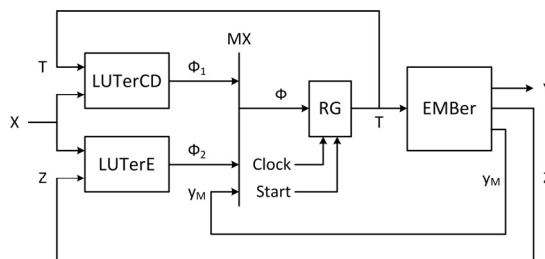


Рис. 2. Структурная схема МПА Мура U_2

Наличие блока *MX* вносит дополнительную задержку в цикл автомата. Поэтому быстродействие схемы $U_2(\Gamma)$ будет ниже, чем схемы $U_1(\Gamma)$. Здесь символ $U_i(\Gamma_j)$ означает, что модель U_i используется для реализации МПА Мура по ГСА Γ_j . Отметим, что окончательный вывод о быстродействии можно сделать только после реализации схем. Возможно, что схемы для Φ_1 и Φ_2 будут иметь меньше уровней, чем схема для Φ . Это может компенсировать временную задержку, вносимую блоком *MX*.

Предлагаемый метод синтеза схемы МПА U_2 включает следующие этапы:

- формирование отмеченной ГСА Γ ;
- формирование разбиения $\Pi_A = \Pi_B \cup \Pi_C$;
- оптимальное кодирование состояний и формирование множеств Π_D и Π_E ;
- кодирование классов $B_i \in \Pi_E$;
- формирование таблицы блока *EMBer*;
- формирование таблицы блока *LUTerCD*;
- формирование таблицы блока *LUTerE*;
- реализация схемы МПА в заданном базисе.

Применение предложенного метода

Пусть для некоторой ГСА Γ_1 множество $A = (a_1, \dots, a_{15})$, $\Pi_A = \{B_1, \dots, B_8\}$, где $B_1 = \{a_1\}$,

$B_2 = \{a_2, a_3, a_4\}$, $B_3 = \{a_5, a_6\}$, $B_4 = \{a_7, a_8, a_9\}$, $B_5 = \{a_{10}, a_{11}, a_{12}\}$, $B_6 = \{a_{13}\}$, $B_7 = \{a_{14}\}$, $B_8 = \{a_{15}\}$. Таким образом, $\Pi_B = \{B_2, B_3, B_4, B_5\}$ и $\Pi_C = \{B_1, B_6, B_7, B_8\}$. Используя методику [5], закодируем состояния $a_m \in A$ оптимальным образом (рис. 3), используя булево пространство размерности $R = 4$, т.е. $T = \{T_1, \dots, T_4\}$.

		T_3T_4			
		00	01	11	10
T_1T_2	00	a_1	a_5	a_6	a_7
	01	a_2	a_3	a_4	*
	11	a_{10}	a_{11}	a_{12}	a_8
	10	a_{13}	a_{14}	a_{15}	a_9

Рис. 3. Оптимальное кодирование состояний автомата $U_2(\Gamma_1)$

Из рис. 3 следует, что $\Pi_D = \{B_2, B_3, B_4\}$ и $\Pi_E = \{B_5\}$, при этом $K(B_2) = 01**$, $K(B_3) = 00*1$, $K(B_4) = **10$, а коды классов $B_i \in \Pi_C$ совпадают с кодами состояний $a_m \in B_i$. В рассматриваемом примере $K(B_1) = 0000$, $K(B_6) = 1000$, $K(B_7) = 1001$, $K(B_8) = 1011$. Итак, $|\Pi_E| = 1$ и из выражения (6) имеем $R_2 = 1$, $Z = \{z_1\}$.

Пусть для ГСА Γ_1 $N = 14$, а для синтеза схемы $EMBer$ используются EMB с $t_F = 4$ при $V = 2^R = 16$. В этом случае $t_S = 4 \cdot 4 = 16$ и $\Delta_t = 16 - 14 = 2$. Таким образом, условие (10) выполняется, и применение предложенного метода имеет смысл. Пусть $K(B_5) = 1$, при этом $z_1 = 0$ свидетельствует о том, что автомат находится в состоянии $a_m \notin B_5$.

Пусть переходы между состояниями автомата $U_2(\Gamma_1)$ заданы следующей системой обобщенных формул перехода [6]:

$$\begin{aligned}
 B_1 &\rightarrow a_2; \\
 B_2 &\rightarrow x_1 a_{10} \vee \overline{x_1} x_2 a_{11} \vee \overline{x_1} \overline{x_2} a_{12}; \\
 B_3 &\rightarrow x_1 a_{13} \vee \overline{x_1} a_{14}; \\
 B_4 &\rightarrow x_1 a_5 \vee \overline{x_1} x_3 a_6 \vee \overline{x_1} \overline{x_3} a_7; \\
 B_5 &\rightarrow x_4 a_2 \vee \overline{x_4} x_3 a_3 \vee \overline{x_4} \overline{x_3} a_4; \\
 B_6 &\rightarrow x_5 a_8 \vee \overline{x_5} a_9; \\
 B_7 &\rightarrow a_{15}; \\
 B_8 &\rightarrow \overline{x_3} a_{10} \vee \overline{x_3} a_{11}.
 \end{aligned} \tag{14}$$

Пусть микрооперации $yn \in Y$ следующим образом распределяются по состояниям авто-

мата $U_2(\Gamma_1)$: $Y(a_1) = \emptyset$, $Y(a_2) = Y(a_6) = \{y_1, y_3\}$, $Y(a_3) = \{y_2, y_4, y_6\}$, $Y(a_4) = Y(a_8) = Y(a_{12}) = \{y_1, y_7, y_8\}$, $Y(a_5) = \{y_3, y_5, y_9\}$, $Y(a_7) = \{y_{10}, y_{11}\}$, $Y(a_9) = \{y_{10}, y_{12}\}$, $Y(a_{10}) = \{y_1, y_{13}, y_{14}\}$, $Y(a_{11}) = Y(a_{15}) = \{y_4, y_{13}\}$, $Y(a_{13}) = \{y_7, y_9\}$, $Y(a_{14}) = \{y_2, y_{12}\}$.

Таблица схемы $EMBer$ включает столбцы a_m , $K(a_m)$, $Y(a_m)$, $K(B_i)$, m , где $K(a_m)$ есть адрес слова EMB . Для автомата $U_2(\Gamma_1)$ эта таблица имеет $V = 16$ строк, которые отображены на рис. 4.

		T_3T_4			
		00	01	11	10
T_1T_2	00	—	Y_3, Y_5, Y_9	Y_1, Y_3	Y_{10}, Y_{11}
	01	Y_1, Y_3	Y_2, Y_4, Y_6	Y_1, Y_7, Y_8, Y_{15}	*
	11	Y_1, Y_3, Y_4, Z_1, Y_M	Y_4, Y_{13}, Z_1, Y_M	$Y_1, Y_7, Y_8, Y_{15}, Z_1, Y_M$	Y_1, Y_7, Y_8
	10	Y_7, Y_9	Y_2, Y_{13}	Y_4, Y_{13}	Y_{10}, Y_{12}

Рис. 4. Содержимое EMB автомата $U_2(\Gamma_1)$

Как видно на рис. 4, переменная $z_1 = 1$ добавлена к наборам микроопераций для состояний $a_m \in B_5$, так как $B_5 \in \Pi_E$. Сюда же добавлена и переменная $y_M = 1$.

Таблицы блоков $LUTerCD$ и $LUTerE$ включают одинаковые столбцы: B_i , $K(B_i)$, a_s , $K(a_s)$, X_h , Φ_h , h , однако для таблицы блока $LUTerCD$ коды $K(B_i)$ представляются переменными $T_r \in T$, а для блока $LUTerE$ — переменными $z_r \in Z$. Часть таблицы для $LUTerCD$ представлена табл. 1, а таблица для $LUTerE$ имеет три строки (табл. 2).

Таблица 1. Фрагмент блока $LUTerCD$ автомата $U_2(\Gamma_1)$

B_i	$K(B_i)$				a_s	$K(a_s)$				X_h	Φ_h	h
	T_1	T_2	T_3	T_4		T_1	T_2	T_3	T_4			
B_1	0	1	*	*	a_{10}	1	1	0	0	X_1	$D_1 D_2$	1
					a_{11}	1	1	0	1	$\overline{X_1} X_2$	$D_1 D_2 D_4$	2
					a_{12}	1	1	1	1	$\overline{X_1} \overline{X_2}$	$D_1 D_2 D_3 D_4$	3
B_6	1	0	0	0	a_8	1	1	1	0	X_5	$D_1 D_2 D_3$	4
					a_9	1	0	1	1	$\overline{X_5}$	$D_1 D_3 D_4$	5

Таблица 2. Блок $LUTerE$ автомата $U_2(\Gamma_1)$

B_i	$K(B_i)$	a_s	$K(a_s)$				X_h	Φ_h	h
			T_1	T_2	T_3	T_4			
B_5	1	a_2	0	1	0	0	X_4	D_2	1
		a_3	0	1	0	1	$\overline{X_4} X_3$	$D_2 D_4$	2
		a_4	0	1	1	1	$\overline{X_4} \overline{X_3}$	$D_2 D_3 D_4$	3

Таблица блока *LUTerCD* используется для формирования системы (11). Например, из табл. 1 можно получить следующие функции (после минимизации):

$$D_1 = \overline{T_1 T_2} \vee T_1 \overline{T_2 T_3 T_4};$$

$$D_2 = \overline{T_1 T_2} \vee T_1 \overline{T_2 T_3 T_4} x_5.$$

Таблица блока *LUTerE* используется для формирования системы (12). Например, следующие функции могут быть получены из табл. 2 (после минимизации):

$$D_2 = z_1;$$

$$D_3 = z_1 \overline{x_4};$$

$$D_4 = z_1 \overline{x_4 x_3}.$$

Последний этап предложенного метода связан с применением пакетов автоматизированного проектирования [7, 8], в статье он не рассматривается.

Заключение. Предложенный метод синтеза основан на наличии свободных выходов встроенных блоков памяти, используемых для реализации системы микроопераций МПА Мура. Это позволяет использовать два источника кодов классов псевдоэквивалентных состояний и удалить из схемы блок преобразователя кодов. Роль этого блока выполняет блок *EMBer* в силу избыточности числа выходов.

Выбор источника кода осуществляется мультиплексором, для управления которым используют переменную u_M . Отметим, что в рассматриваемом примере можно использовать переменную z_1 . Если блок *EMBer* не имеет достаточно свободных выходов, то переменная u_M может быть реализована на *LUT*-элементах как функция

$$u_M = f(Z). \quad (15)$$

Исследования, проведенные для стандартных примеров из библиотеки [12] показали, что при выполнении условия (10) МПА $U_2(\Gamma)$ всегда имеет меньше *LUT*-элементов, чем эквивалентный МПА $U_1(\Gamma)$. Максимальный выигрыш может достигать 32 процента. При этом для 94 процентов стандартных примеров блок *EMBer* реализуется в виде одного блока *EMB*.

Кроме того, в 86 процентах рассмотренных МПА из библиотеки [12] быстродействие моделей $U_1(\Gamma)$ и $U_2(\Gamma)$ совпадает. В оставшихся 14 процентах примеров автомат $U_2(\Gamma)$ обладает несколько меньшим быстродействием (снижается на значение около 10 процентов).

Научная новизна предложенного метода заключается в использовании особенностей МПА Мура и встроенных блоков памяти микросхем *FPGA* для уменьшения числа *LUT*-элементов в схеме МПА.

Практическая значимость метода заключается в уменьшении площади кристалла *FPGA*, занимаемой схемой автомата Мура, в сравнении с известными аналогами.

1. Grout I. Digital Systems Design with FPGAs and CPLDs. – Amsterdam: Elsevier, 2008. – 328 p.
2. Skliarova I., Skliarov V., Sudnitsion A. Design of FPGA-based Hierarchical Finite State Machines. – Tallinn: TUT Press, 2012. – 286 p.
3. De Micheli G. Synthesis and Optimization of Digital Circuits. – New York: McGraw Hill, 1994. – 636 p.
4. Baranov S. Logic Synthesis for Control Automata. – New York: Kluwer Acad. Publ., 1994. – 312 p.
5. Баркалов А.А. Принципы оптимизации логической схемы микропрограммного автомата Мура. // Кибернетика и системный анализ. – 1998. – № 1. – С. 65–72.
6. Barkalov A., Titarenko L. Logic Synthesis for FSM-based Control Units. – Berlin: Springer, 2009. – 234 p.
7. Altera Corporation. FPGA, CPLID and Structured ASIC. – <http://www.altera.com>.
8. Xilinx Inc. FPGA, CPLD Solutions. – <http://www.xilinx.com>.
9. Scholl C. Functional Decomposition with Application to FPGA Synthesis. – Norwell: Kluwer Acad. Publ., 2001. – 319 p.
10. Баркалов А.А., Титаренко Л.А., Цололо С.А. Оптимизация схемы автомата Мура, реализуемой в базисе ПЛИС // Кибернетика и системный анализ. – 2009. – № 5. – С. 180–186.
11. Баркалов А.А., Цололо С.А. Оптимизация автомата Мура, реализуемого в базисе *CPLD* // УСИМ. – 2008. – № 4 – С. 43–48.
12. Yang S. Logic Synthesis and Optimization Benchmarks user guide. Technical Report. – North Carolina: MCNC, 1991. – 44 p.

Поступила 10.06.2013

Тел. для справок: +38 062 301-0723, (Донецк)

E-mail: A.Barkalov@jie.uz.zgora.pl, s.solos@gmail.com,

MiroshkinAN@gmail.com

© А.А. Баркалов, Л.А. Титаренко, С.А. Цололо, А.Н. Мирошкин, 2013